

Федеральное государственное бюджетное образовательное учреждение высшего образования



**«Московский государственный технический  
университет  
имени Н.Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ ФУНДАМЕНТАЛЬНЫЕ НАУКИ  
КАФЕДРА МАТЕМАТИКА И КОМПЬЮТЕРНЫЕ НАУКИ

### Отчет

по лабораторному заданию № 12

Вариант 18

Дисциплина: Информатика


Название лабораторного задания: Использование указателей при работе с массивами в C++. Динамические двумерные массивы.

Студент гр. ФН11-22Б

ХС 02.03.21  
(Подпись, дата)

**М.Х. Хаписов**  
(И.О. Фамилия)

Преподаватель Доцент кафедры ИУ-6

3.03.21   
(Подпись, дата)

**Т.Н. Ничушкина**  
(И.О. Фамилия)

Москва, 2021

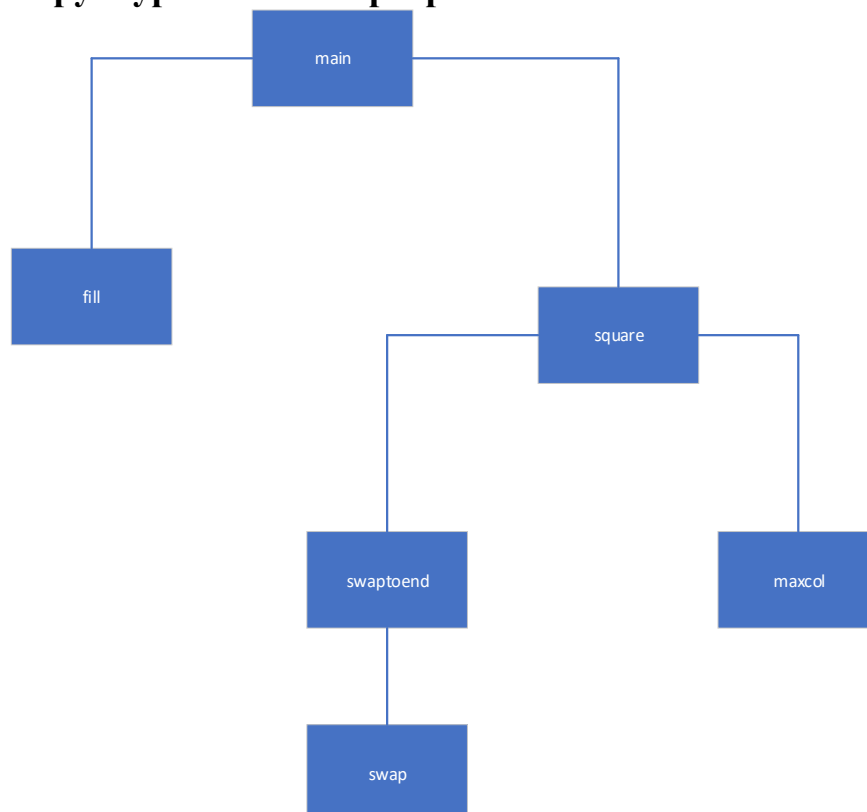
## Цель:

- Приобретение навыков работы с динамической памятью и указателями на языке программирования C++.
- Изучение принципов и приемов работы с динамическими многомерными массивами.

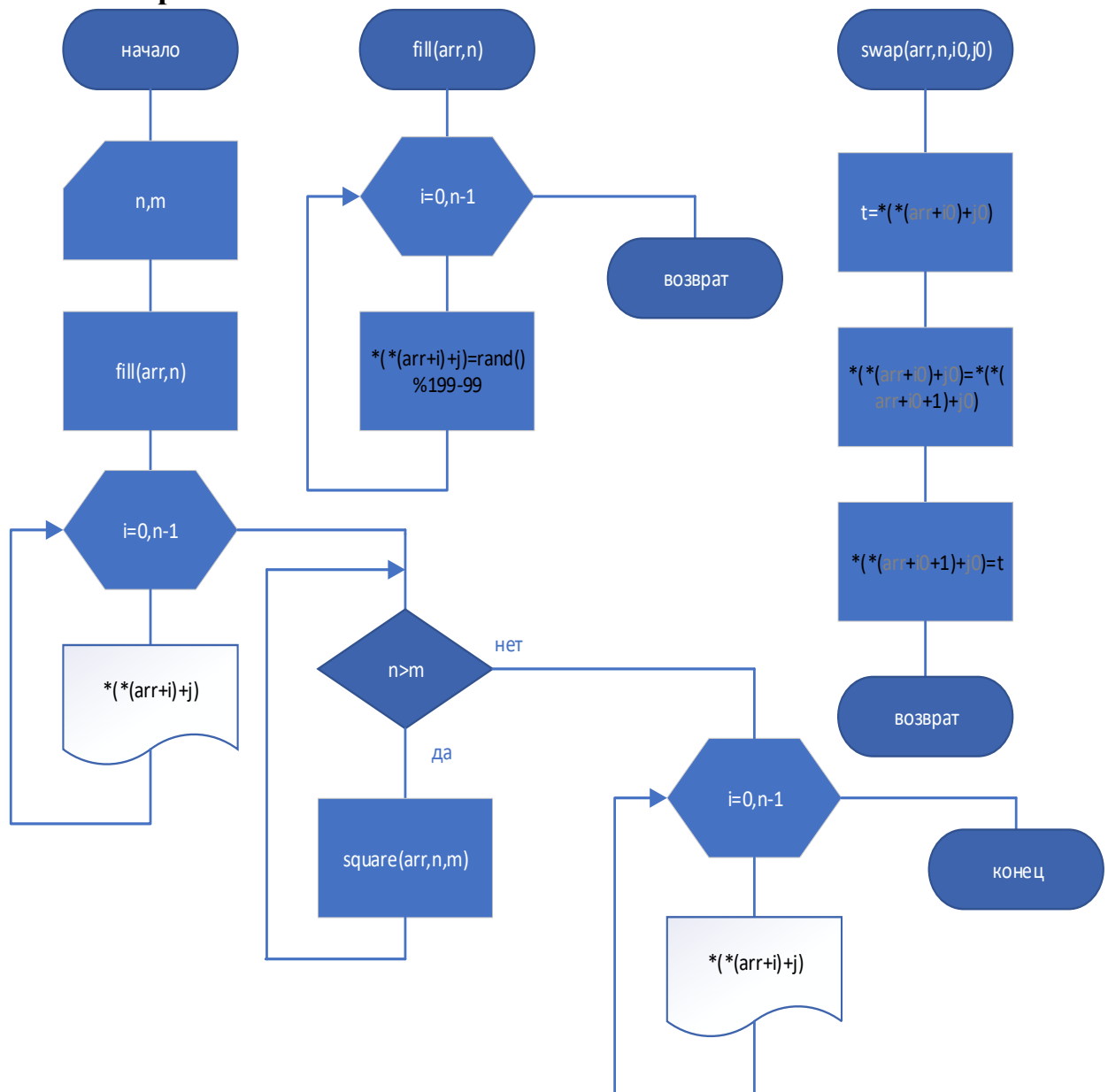
**Задание. а)** Сформировать в динамической памяти целочисленную матрицу размером  $N \times M$  ( $N > M$ ;  $N, M$  вводятся с клавиатуры). Получить квадратную матрицу порядка  $M$  путем удаления наибольших элементов каждого столбца, сохраняя при этом порядок остальных. Пользуясь **указателями**, вывести на экран адреса найденных максимальных элементов. Вывести на экран исходную и полученную матрицы.

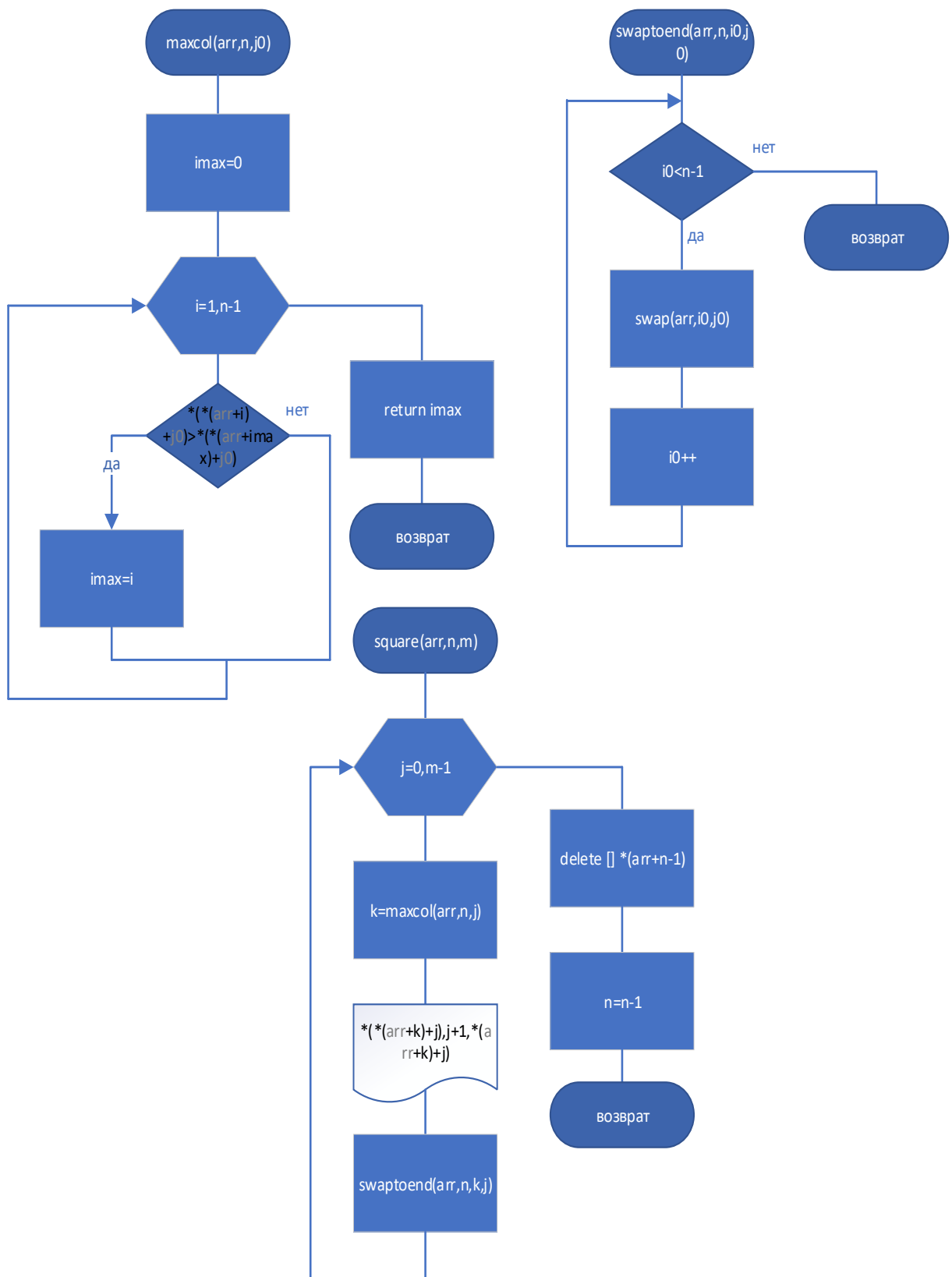
**б)** Решить поставленную задачу, используя средства управления вводом/выводом C++. Сформировать квадратную целочисленную матрицу, размером не более  $10 \times 10$ . Полученную матрицу вывести на экран по строкам, смещая все нечетные строки вправо

## Структурная схема программы



## Схема алгоритма





## Текст алгоритма

```
#include <iostream>
#include <ctime>
void fill(int **arr,int n,int m) {
    srand(time(NULL));
    for(int i=0;i<n;i++)
        for(int j=0;j<m;j++) *(arr+i+j)=rand()%199-99;
}
int maxcol(int **arr,int n,int j0) {
    int imax=0;
    for(int i=1;i<n;i++)
        if(*(arr+i+j0)>*(arr+imax+j0)) imax=i;
    return imax;
}
void swap(int **arr,int i0,int j0) {
    int t=*(arr+i0+j0);
    *(arr+i0+j0)=*(arr+i0+1+j0);
    *(arr+i0+1+j0)=t;
}
void swaptocend(int **arr,int n,int i0,int j0) {
    while(i0<n-1) {
        swap(arr,i0,j0);
        i0++;
    }
}
void square(int **arr,int &n,int m) {
    for(int j=0;j<m;j++) {
        int k=maxcol(arr,n,j);
        printf("Address of %d (max in %d column) is %p\n",*(arr+k+j),j+1,*(arr+k+j));
        swaptocend(arr,n,k,j);
    }
    delete[] *(arr+n-1);
    n--;
}
void main() {
    int n,m;
    puts("Input size of array");
    std::cin>>n>>m;
    while((n<=0)|| (m<=0)|| (n<=m)) {
        puts("Error! Incorrect data! Inputs size of array again");
        std::cin>>n>>m;
    }
    int **arr=new int*[n];
    for(int i=0;i<n;i++) *(arr+i)=new int[m];
    fill(arr,n,m);
    puts("Initial array:");
    for(int i=0;i<n;i++) {
        for(int j=0;j<m;j++) printf("%3d ",*(arr+i+j));
        puts("");
    }
    while(n>m) square(arr,n,m);
    puts("Transformed array:");
    for(int i=0;i<n;i++) {
        for(int j=0;j<m;j++) printf("%3d ",*(arr+i+j));
        puts("");
    }
    puts("With shifts:");
    for(int i=0;i<n;i++) {
        if(i%2!=0) printf(" ");
        for(int j=0;j<m;j++) printf("%3d ",*(arr+i+j));
        puts("\n");
    }
}
```

```

    for (int i=0;i<n;i++) delete[] arr[i];
    system("pause");
}

```

## Тестирование

```

C:\Users\Knigan\source\repos\Project4\Debug\Project4.exe
Input size of array
0 0
Error! Incorrect data! Inputs size of array again
0 1
Error! Incorrect data! Inputs size of array again
1 0
Error! Incorrect data! Inputs size of array again
1 1
Error! Incorrect data! Inputs size of array again
2 1
Initial array:
 97
-22
Address of 97 (max in 1 column) is 0108D688
Transformed array:
-22
With shifts:
-22

Для продолжения нажмите любую клавишу . . .

C:\Users\Knigan\source\repos\Project4\Debug\Project4.exe
Input size of array
2 3
Error! Incorrect data! Inputs size of array again
3 2
Initial array:
 9 68
-2 -55
-13 -70
Address of 9 (max in 1 column) is 0089E438
Address of 68 (max in 2 column) is 0089E43C
Transformed array:
-2 -55
-13 -70
With shifts:
-2 -55

      -13 -70

Для продолжения нажмите любую клавишу . . .

```

C:\Users\Knigan\source\repos\Project4\Debug\Project4.exe

Input size of array

11 10

Initial array:

```
-56 43 -9 -14 85 71 -20 25 2 -32
-51 77 -39 48 -1 -27 55 66 -38 -71
-93 95 -49 4 -11 57 87 -19 69 67
46 -33 -43 27 -64 9 -74 15 -41 -99
-65 -4 24 -86 18 45 -26 56 34 -55
-85 41 -64 34 -12 -26 24 79 6 81
20 -10 -47 73 81 18 -95 -49 -84 -4
-25 28 -59 34 -22 -71 35 6 -4 57
-49 -63 32 33 -52 95 -9 -3 -15 7
-35 -7 -52 -54 13 95 2 62 -66 62
3 -65 25 -79 -78 78 -78 84 -41 56
```

Address of 46 (max in 1 column) is 00901238

Address of 95 (max in 2 column) is 008FD0CC

Address of 32 (max in 3 column) is 00901DC0

Address of 73 (max in 4 column) is 00901E1C

Address of 85 (max in 5 column) is 008F51A0

Address of 95 (max in 6 column) is 00901DCC

Address of 87 (max in 7 column) is 008FD0E0

Address of 84 (max in 8 column) is 00901BC4

Address of 69 (max in 9 column) is 008FD0E8

Address of 81 (max in 10 column) is 0090209C

Transformed array:

```
-56 43 -9 -14 -1 71 -20 25 2 -32
-51 77 -39 48 -11 -27 55 66 -38 -71
-93 -33 -49 4 -64 57 -74 -19 -41 67
-65 -4 -43 27 18 9 -26 15 34 -99
-85 41 24 -86 -12 45 24 56 6 -55
20 -10 -64 34 81 -26 -95 79 -84 -4
-25 28 -47 34 -22 18 35 -49 -4 57
-49 -63 -59 33 -52 -71 -9 6 -15 7
-35 -7 -52 -54 13 95 2 -3 -66 62
3 -65 25 -79 -78 78 -78 62 -41 56
```

With shifts:

```
-56 43 -9 -14 -1 71 -20 25 2 -32
-51 77 -39 48 -11 -27 55 66 -38 -71
-93 -33 -49 4 -64 57 -74 -19 -41 67
-65 -4 -43 27 18 9 -26 15 34 -99
-85 41 24 -86 -12 45 24 56 6 -55
20 -10 -64 34 81 -26 -95 79 -84 -4
-25 28 -47 34 -22 18 35 -49 -4 57
-49 -63 -59 33 -52 -71 -9 6 -15 7
-35 -7 -52 -54 13 95 2 -3 -66 62
3 -65 25 -79 -78 78 -78 62 -41 56
```

Для продолжения нажмите любую клавишу . . .

```
C:\Users\Knigan\source\repos\Project4\Debug\Project4.exe
Input size of array
8 5
Initial array:
 87 -40 -20 47 -50
 52 85 -91 44 -45
  8 92 -13 26 -19
 98  1 66 52 -97
-77 93 70  4 27
 98 60 35 44  3
 21 -57 82 43 -4
-92 35 25 52 93
Address of 98 (max in 1 column) is 012252D0
Address of 93 (max in 2 column) is 01225314
Address of 82 (max in 3 column) is 01220588
Address of 52 (max in 4 column) is 012252DC
Address of 93 (max in 5 column) is 01230F08
Address of 98 (max in 1 column) is 01225310
Address of 92 (max in 2 column) is 012251D4
Address of 70 (max in 3 column) is 01225318
Address of 52 (max in 4 column) is 0122058C
Address of 27 (max in 5 column) is 01225320
Address of 87 (max in 1 column) is 012254A0
Address of 85 (max in 2 column) is 01225194
Address of 66 (max in 3 column) is 012252D8
Address of 47 (max in 4 column) is 012254AC
Address of 3 (max in 5 column) is 01225320
Transformed array:
 52 -40 -20 44 -50
  8  1 -91 26 -45
-77 60 -13  4 -19
 21 -57 35 44 -97
-92 35 25 43 -4
With shifts:
 52 -40 -20 44 -50

    8  1 -91 26 -45

-77 60 -13  4 -19

    21 -57 35 44 -97

-92 35 25 43 -4

Для продолжения нажмите любую клавишу . . .
```

Расстояние между элементами 93 и 70 (максимумы в одной строке), которые к тому же являются соседними, составляет  $|01225314 - 01225318| = 4_{16} = 4_{10}$ , что говорит о том, что в памяти они также располагаются друг за другом.

Расстояние между элементами 70 и 66 (максимумы в одном столбце) составляет  $|01225318 - 012252D8| = 40_{16} = 64_{10}$ , что говорит о том, что элементы



массива располагаются в памяти друг за другом только в рамках одной строки, а при переходе на новую строку адрес элементов может резко измениться.

**Вывод:** я приобрёл некоторые навыки работы с динамической памятью и указателями на языке программирования C++ и изучил принципы и приёмы работы с динамическими многомерными массивами

### **Ответы на контрольные вопросы:**

1. Что такое динамический многомерный массив? (Ответ: динамический многомерный массив – это многомерный массив, размер которого может изменяться во время исполнения программы. Возможность изменения размера отличает динамический массив от статического, размер которого задаётся на момент компиляции программы)

1. Как динамически выделяется память под динамический многомерный массив?

```
short **matr;
```

2. Как выполнить доступ к элементам динамического массива?

```
D2[1][2] ⇔ * (* (D2+1) +2)
```

3. Как освобождается память, выделенная для динамического массива?

```
for(i=0;i<n;i++) // удаление строк динамической матрицы
    delete[] mas[i];
delete[] mas; //удаление массива указателей на строки
```